⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平3-113563

| ⑤Int. Cl.⁵ | 識別記号 | 庁内整理番号 | ⑬公開 平成3年(1991)5月14日 |
|---|---|---|---|
| G 06 F 15/16 | 380 Z | 6945-5B | |

審査請求 未請求 請求項の数 16 (全18頁)

⑭発明の名称　マルチプロセッサスケジューリング方法

⑳特　願 平1-250532
㉒出　願 平1(1989)9月28日

㉑発 明 者　上　脇　　正　茨城県日立市久慈町4026番地　株式会社日立製作所日立研究所内
㉑発 明 者　山　口　伸一朗　茨城県日立市久慈町4026番地　株式会社日立製作所日立研究所内
㉑発 明 者　斉　藤　雅　彦　茨城県日立市久慈町4026番地　株式会社日立製作所日立研究所内
㉑発 明 者　小　林　芳　樹　茨城県日立市久慈町4026番地　株式会社日立製作所日立研究所内
㉑出 願 人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
㉑代 理 人　弁理士 秋本 正実

最終頁に続く

明　細　書

1．発明の名称

マルチプロセッサスケジューリング方法

2．特許請求の範囲

1．複数のプロセッサを具備し、該複数のプロセッサが、単一のオペレーティングシステムによって制御されるマルチプロセッサシステムにおいて、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラが、プロセスに何れかのプロセッサを割り当てるプロセススケジューリング方式であって、上記プロセス管理テーブルに属性を許すグループ識別子を記憶し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を記憶し、上記プロセススケジューラは、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのプロセッサ割り当て情報により指定されたプロセッサ上で該プロセスを動作させることにより、同一のグループに属するプロセスは、同一のプロセッサで実行することを特徴とするマルチプロセッサスケジューリング方法。

2．上記プロセス管理テーブルに該プロセスの実行待ち時間を記憶し、上記プロセススケジューラが次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、実行待ち時間がある値以上の時には、任意のプロセッサで動作させ、該実行待ち時間がある値以下の時には、該プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのプロセッサ割り当て情報により指定されたプロセッサ上で該プロセスを動作させることを特徴とする請求項1のマルチプロセッサスケジューリング方法。

3．上記グループ管理テーブルに該グループの何れのプロセスも実行されていない時間、即ちグ

ループ実行待ち時間を記憶し、上記プロセススケジューラが、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのグループ実行待ち時間がある値以上の時には、該グループ管理テーブルのプロセッサ割り当て情報を変更できるようにしたことを特徴とするマルチプロセッサスケジューリング方法。

4．上記プロセス管理テーブルに該プロセスの実行待ち時間と該プロセスが前回実行されたプロセッサの識別子を記憶し、上記プロセススケジューラが、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、該待ち時間がある値以下の時には、該プロセスを前回実行したプロセッサか、グループ管理テーブルのプロセッサ割り当て情報が指示するプロセッサで実行し、ある値以上の時には、上記以外のプロセッサで動作できるようにしたことを特徴とする請求項1又は2のマルチプロセッサスケジューリング方法。

5．該プロセスの実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いたことを特徴とする請求項2又は3又は4のマルチプロセッサスケジューリング方法。

6．該グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いたことを特徴する請求項3又は5のマルチプロセッサスケジューリング方法。

7．複数のプロセッサを具備し、該複数のプロセッサが、単一のオペレーティングシステムによって制御されるマルチプロセッサシステムにおいて、実行しているプロセスを切り替える必要が生じたプロセッサが、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラを実行し、該プロセッサが次に実行するプロセスを決定するプロセススケジューリング方式であって、上記プロセス管理テーブルに登録を許すグループ識別子を記述し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を記述し、上記プロセススケジューラは、実行可能プロセスの中から、上記プロセス管理テーブルのグループ識別子が指示する、該グループ管理テーブルのプロセッサ割り当て情報が該プロセススケジューラを実行しているプロセッサを指示しているプロセスを次に実行することを特徴とするマルチプロセッサスケジューリング方法。

8．上記プロセス管理テーブルに該プロセスの実行待ち時間を記憶し、上記プロセススケジューラが、次に実行するプロセスを選択する際に上記プロセス管理テーブルを参照して、実行待ち時間がある値以上の時には、該プロセスを実行可能とすることを特徴とする請求項7のマルチプロセッサスケジューリング方法。

9．上記グループ管理テーブルに該グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間を記憶し、上記プロセススケジューラが、次に実行するプロセスを選択する際に、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのグループ実行待ち時間がある値以上の時には、該グループ管理テーブルのプロセッサ割り当て情報を変更して、該プロセスを実行可能とすることを特徴とする請求項7又は8のマルチプロセッサスケジューリング方法。

10．上記プロセス管理テーブルに該プロセスの実行待ち時間と該プロセスが前回実行されたプロセッサの識別子を記憶し、上記プロセススケジューラが、次に実行するプロセスを選択する際に、上記プロセス管理テーブルを参照して、該待ち時間がある値以下の時には、該プロセスを前回実行したプロセッサか、グループ管理テーブルのプロセッサ割り当て情報が指示するプロセッサが該プロセススケジューラを実行してい

るプロセッサと同一である場合にのみ実行可能とし、該待ち時間がある値以上の時には、実行可能とすること、 を特徴とする請求項7のマルチプロセッサスケジューリング方法。

11. 該プロセスの実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いたことを特徴とする請求項8又は9又は10のマルチプロセッサスケジューリング方法。

12. 該グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いたことを特徴とする請求項8又は6のマルチプロセッサスケジューリング方法。

13. 複数のプロセッサを具備し、該複数のプロセッサが、単一のオペレーティングシステムによって制御されるマルチプロセッサシステムにおいて、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラが、プロセスに何れかのプロセッサを割り当てるプロセススケジューリング方式であって、上記プロセス管理テーブルに上記複数のプロセッサ各々に対する優先度と重視を許すグループ識別子を記述し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を設け、上記プロセス管理テーブルのグループ識別子が示すグループ管理テーブル中のプロセッサ割り当て情報が指示するプロセッサに対応する該優先度を高く設定することを特徴とするマルチプロセッサスケジューリング方法。

14. プロセス割り当て情報のコードの一つとして、該プロセスが何れのプロセッサでも実行可能であることを示すコードを設けたことを特徴とする請求項1～13のいずれか1つに記載のマルチプロセッサスケジューリング方法。

15. プロセス管理テーブル、グループ管理テーブルの何れか、または、両方に該プロセスが特定のプロセッサでのみ実行可能であることを示すフラグを設けたことを特徴とする請求項1～14のいずれか1つに記載のマルチプロセッサスケジューリング方法。

16. 資源を共有するプロセスに対して、同一のグループ識別子を持たせることにより、該資源の排他制御を簡単化したことを特徴とする請求項1～15のいずれか1つに記載のマルチプロセッサスケジューリング方法。

3. 発明の詳細な説明

【産業上の利用分野】

複数のプロセッサから成るマルチプロセッサシステムで複数のプロセスを並行に動作させるマルチプロセッサ用オペレーティングシステムにおいて、特に、高速システムを構築する場合、あるいはシングルプロセッサ用に作成されたプログラムをマルチプロセッサシステムで、なんら変更なく動作させる場合に好適なプロセススケジューリング方法に関する。

【従来の技術】

複数のマイクロプロセッサを使用したマルチマイクロプロセッサの開発が進んである。特開昭55-127261号、マルチプロセッサシステムはこの一例であり、複数のプロセッサがバスにより結合されたシステム構成を示している。マルチプロセッサ用オペレーティングシステムについても盛んに開発が行なわれている。マルチプロセッサ用オペレーティングシステムにおいてはシングルプロセッサシステムとは異なり、複数のプロセスが異なるプロセッサ上で同時に実行できるようにする必要があり、このために、プロセスをどのプロセッサに割り当てるかを決定する方策を持つ必要がある。この場合、最も一般的に行なわれているのはプロセッサの利用効率を最大限にするために、負荷の最も少ないプロセッサに割り当てるという方策である。この方策は一般的に実現が容易であるため、広く採用されている。というのも、個々のプロセッサ上で動作するオペレーティングシステムのプロセススケジューラがそれぞれシングルプロセッサシステムの場合と同様に次に実行すべきプロセスをプロセスの優先順位に従って選択すれ

ばよいからである。負荷の重いプロセッサではプロセスの実行時間が長く、プロセススケジューラが起動される頻度は低い。逆に負荷の軽いプロセッサでは処理すべきプロセスを持たないアイドル状態になるタイミングが多く、したがってアイドル状態に移行すべきかどうかを判定するプロセススケジューラの起動タイミングも多い。このため、負荷の軽いプロセッサにはより多くのプロセススケジューリングの機会が与えられ、プロセッサ間での負荷は自動的に均等になるように推移するのである。以上の方策は機種が同じ均質なマルチプロセッサシステムにおいては合理性のあるプロセスの割り当て方針である。

また、プロセスをグループとして管理して、スケジューリングする方法について、「エクスペアリンス・ユーズィング：マルチープロセッサ・システムズーAスティタス・レポート」（"Experience Using Multi-processor Systems-A Status Report"）（ACM Computing Surveys, 12巻2号p121～165, 1980）にグループスケジューリング（group scheduling）として記されている。これは、並列処理可能なプロセスをグループとして、スケジューリングするときにはこれらのプロセスをできるかぎり同時に異なるプロセッサで実行するようにする。

【発明が解決しようとする課題】

複数のプロセスをグループとし、同一グループに属するプロセスをできるかぎり同一のプロセッサで実行する機能を有するプロセススケジューラが好ましい。

ここでプロセスをグループ化するのは以下のようなユーザ要求およびハードウェアならびにオペレーティングシステム構築上の要求があるためである（尚、このグループとは同じデータを扱うプロセスの集まりである）。

共有バス、共有メモリを持ったマルチプロセッサでは、バスの負荷を軽くするため、各プロセッサがキャッシュメモリを持つ。この様なシステムでは、キャッシュ間でデータの一貫性を保証するために一つのプロセッサがキャッシュ上のデータ





を更新すると、それと同じデータを持っているキャッシュに更新した結果を転送する。お互いに多くのデータを共有しているプロセスを異なるプロセッサで実行するとキャッシュ間のデータ転送の回数が増大し、キャッシュの性能が発揮されず、バスの負荷が高くなってしまう；従来のマルチプロセッサシステムでは、必ずしもこの問題点に関して十分な配慮がなされていなかった。

また、シングルプロセッサ用に開発された既存のソフトウェアは、マルチプロセッサシステム上においても何等変更を加えずに実行させることが望ましい。マルチプログラミング環境では、周知のように、複数のプロセスの競合が発生しうる。この問題を解決するため、通常、共有データにアクセスする際にセマフォメカニズム等により排他制御を行なう必要があるが、セマフォメカニズムの処理オーバヘッドが問題となり、プロセスのプライオリティを同一にしたり、セマフォメカニズムよりも処理時間の短いプロセッサ割り込みマスクを上げて割り込み禁止にする等の手段を講じる場合が多い。これらの手段は、プロセッサ1台上でミクロに見ればプロセッサ上の処理は必ず逐次的に行なわれるという事実を利用している。現実には、共有データに対する排他制御を性能上の理由により行なっていないソフトウェアも多い。このため既存のソフトウェアをその様マルチプロセッサシステムで実行させた場合、必ずしも正常な動作に補償されないという問題があった。

従来のgroup schedulingは、グループとなっているプロセッサと異なるプロセッサで動作させているため、並列処理には適しているが、上記のような問題には対応できなかった。

尚、本願に関係する先願は、同一出願人になる特願平1－60091号「マルチプロセッサ用オペレーティングシステム」がある。この先願はプロセス管理テーブルを持ち、このテーブルにプロセス占有プロセッサ名を記録しておき、他プロセッサ支配下のプロセスを一定条件のもとに横取りして自己の管理において処理を行なわせることとしている。この先願には、前回使用プロセッサの記録

やグループ化のプロセス管理の記載はない。

本発明の目的は、マルチプロセッサシステムにおいて、プロセスをグループ化して管理し、同一グループ内のプロセスを同一プロセッサで実行することにより、キャッシュメモリの性能を向上させ、バスの負荷を低減できるプロセススケジューリング方法を提供することにある。

本発明の別の目的は、既存の単一プロセッサシステム用のソフトウェアを、なんら変更させずそのまま利用できるプロセススケジューリング方法を提供することにある。

本発明の更に別の目的は、グループとなったプロセスを同一プロセッサで実行する際にも、プロセッサ間で仕事の負荷を均等に保つことができるプロセススケジューリング方法を提供することにある。

【課題を解決するための手段】

上記目的を達成するために、本発明によるプロセススケジューリング方法は、複数のプロセッサを具備し、上記複数のプロセッサが、単一のオペレーティングシステムによって制御されるマルチプロセッサシステムにおいて、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラが、プロセスに何れかのプロセッサを割り当てるプロセススケジューリング方式であって、上記プロセス管理テーブルに登録を許すグループ識別子を用意し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を記述し、上記プロセススケジューラは、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのプロセッサ割り当て情報により指定されたプロセッサ上で該プロセスを動作させることにより、同一のグループに属するプロセスは、同一のプロセッサで実行するようにしたものである。

上記プロセス管理テーブルに該プロセスの実行





待ち時間を記憶して、上記プロセススケジューラが、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、実行待ち時間がある値以上の時には、任意のプロセッサで動作させ、該実行待ち時間がある値以下の時には、該プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのプロセッサ割り当て情報により指定されたプロセッサ上で該プロセスを動作させるようにしてもよい。

さらに上記グループ管理テーブルに該グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間を記憶し、上記プロセススケジューラが、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのグループ実行待ち時間がある値以上の時には、該グループ管理テーブルのプロセッサ割り当て情報を変更できるようにしてもよい。

さらに、上記プロセス管理テーブルに該プロセスの実行待ち時間と該プロセスが前回実行されたプロセッサの識別子を記憶し、上記プロセススケジューラが、次に実行すべく選択したプロセスについて、上記プロセス管理テーブルを参照して、該待ち時間がある値以下の時には、該プロセスを前回実行したプロセッサか、グループ管理テーブルのプロセッサ割り当て情報が指示するプロセッサで実行し、ある値以上の時には、上記以外のプロセッサで動作できるようにしてもよい。

さらに、上記プロセスの実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いてもよい。

さらに、上記グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いた。

本発明の他のプロセススケジューリング方法として、複数のプロセッサを具備し、該複数のプロセッサが、単一のオペレーティングシステムによ

って制御されるマルチプロセッサシステムにおいて、実行しているプロセスを切り替える必要が生じたプロセッサが、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラを実行し、該プロセッサが次に実行するプロセスを決定するプロセススケージューリング方式であって、上記プロセス管理テーブルに直接を許すグループ識別子を記述し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を記述し、上記プロセススケジューラは、実行可能プロセスの中から、上記プロセス管理テーブルのグループ識別子が指示する、該グループ管理テーブルのプロセッサ割り当て情報が該プロセススケジューラを実行しているプロセッサを指示しているプロセスを次に実行するようにしてもよい。

上記プロセス管理テーブルに該プロセスの実行待ち時間を記憶し、上記プロセススケジューラが、



実行したプロセッサか、グループ管理テーブルのプロセッサ割り当て情報が指示するプロセッサが該プロセススケジューラを実行しているプロセッサと同一である場合にのみ実行可能とし、該待ち時間がある値以上の時には、実行可能としてもよい。

さらに、上記プロセスの実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いてもよい。

さらに、上記グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間の代わりに該プロセスのプロセッサへの割り当てが拒絶された回数を用いた。

本発明の他のプロセススケジューリング方式として、複数のプロセッサを具備し、該複数のプロセッサが、単一のオペレーティングシステムによって制御されるマルチプロセッサシステムにおいて、プロセスを管理するプロセス管理テーブルに従い、上記オペレーティングシステムのプロセススケジューラが、プロセスに何れかのプロセッサ

次に実行するプロセスを選択する際に上記プロセス管理テーブルを参照して、実行待ち時間がある値以上の時には、該プロセスを実行可能ととしてもよい。

さらに、上記グループ管理テーブルに該グループの何れのプロセスも実行されていない時間、即ちグループ実行待ち時間を記憶し、上記プロセススケジューラが、次に実行するプロセスを選択する際に、上記プロセス管理テーブルを参照して、当該プロセスのグループを判別し、該グループのグループ管理テーブルのグループ実行待ち時間がある値以上の時には、該グループ管理テーブルのプロセッサ割り当て情報を変更して、該プロセスを実行可能としてもよい。

さらに、上記プロセス管理テーブルに該プロセスの実行待ち時間と該プロセスが前回実行されたプロセッサの識別子を記憶し、上記プロセススケジューラが、次に実行するプロセスを選択する際に、上記プロセス管理テーブルを参照して、該待ち時間がある値以下の時には、該プロセスを前回



を割り当てるプロセススケジューリング方式であって、上記プロセス管理テーブルに上記複数のプロセッサ各々に対する優先度と直接を許すグループ識別子を記述し、該グループ識別子が同一な該プロセスの情報を記憶するグループ管理テーブルを設け、該グループ管理テーブルにグループ内のプロセスを実行するプロセッサを指示するプロセッサ割り当て情報を設け、上記プロセス管理テーブルのグループ識別子が示すグループ管理テーブル中のプロセッサ割り当て情報が指示するプロセッサに対応する該優先度を高く設定するようにしてもよい。

【作用】

本発明のプロセススケジューリング方法は、複数のプロセッサからなるマルチプロセッサシステムにおいて、複数のプロセッサのいずれでも実行可能なプロセスについて、あえてそのプロセスをグループ化しそのグループを実行させるプロセッサを特定のプロセッサに限定しようとするものである。

共有データを多く持つプロセスが異なるプロセッサで実行されると、キャッシュ一致化のための通信が頻繁に発生しキャッシュ性能を低下させる。本発明では、そのようなプロセスをグループ化し、同一のプロセッサで実行できるので、キャッシュ一致化のための通信が不要となり、性能を向上させることができる。さらに、グループを特定のプロセッサに実行させることによりプロセッサ間の負荷が片寄ることがないように、一定時間以上実行されずに放置されているプロセスについては、いずれのプロセッサでも実行できるようにしており、負荷分散が可能となっている。ただし、この機能はプロセス固定フラグを立てることにより禁止することもできる。

また、本発明によって、排他制御を必要とするプロセス群を同一のプロセッサで動作させることが可能になり、共有メモリに対する排他制御を考慮していない従来の単一プロセッサ用の既存のプロセスを修正することなく使用しても正常な動作を保証することができる。この事は、他のプロセッサの不使用を意味するものではなく、前記プロセス群以外のプロセスについては、このような制約を設けずに、任意のプロセッサで実行可能としてマルチプロセッサの機能を有効に利用することができる。

【実施例】

以下、本発明の一実施例を第１図を用いて説明する。第１図において、13－１～13－２はプロセッサ、12－１～12－６はプロセス、15－１～15－３はグループである。

プロセッサ13－１～13－３は、プロセス12－１～12－６を切り替えながら実行する。プロセス12－１～12－３はグループ15－１として、プロセス12－４と12－５はグループ15－２として、プロセス12－６～12－８はグループ15－３として、まとめられている。プロセスをグループにする方法の指示は、ユーザによってなされるが、コンパイラやオペレーティングシステムが自動的に行なっても良い。

プロセスを実行するプロセッサはグループ毎に





決まっており、同一のグループに属するプロセスは、同一のプロセッサで実行するようにする。第１図ではグループの数とプロセッサの数が等しくなっているが、これらの数は、特に一致している必要はなく、どちらが多くても少なくても良い。

本発明によるシステム構成を第２図に示す。第２図において、13－１～13－３はプロセッサ、11－１～11－３はプロセッサ番号、12－１～12－６はプロセス、14－１はタイマー、４は接続バス、５は共有メモリ、31はＯＳプログラム、311はスケジューラ、32－１～32－４はプロセス管理テーブル、321－１～321－４はプロセッサ識別子、322－１～322－４は優先度、323－１～323－４は実行待ち時間、324－１～324－４はカウンタ、325－１～325－４はプロセス固定フラグ、326－１～326－４はグループ識別子、33－１～33－２はグループ管理テーブル、331－１～331－２はプロセッサ識別子、332－１～332－２は実行待ち時間、333－１～333－２はカウンタ、334－１～334－２はグループ固定フラグである。

プロセッサ13－１～13－３は、接続バス４により相互に、ならびに共有メモリ５に接続されている。各プロセッサはプロセッサ番号11－１～11－３をそれぞれ持っている。本実施例では、プロセッサ13－１～13－３はそれぞれプロセッサ番号"１"，"２"，"３"を有している。このプロセッサ番号の設定は、本実施例ではハードウェアスイッチにより行なっているが、他の方法として、ハードのＲＯＭにあらかじめ記憶しておく、立ち上げ時にプログラムにより設定する等の方法も考えられる。

プロセッサ13－１～13－３のうち、一つのプロセッサにはタイマー14－１があり、一定の時間おきにタイマー割り込みがそのプロセッサに入る。

各プロセスはプロセッサ上で実行される。共有メモリ５には、オペレーティングシステムのプログラム31、このプログラム31の一部であるプロセススケジューラ311、このオペレーティングシステムが管理するテーブルの一部であるプロセス管理テーブル32（32－１～32－３）、グループ管理

テーブル33（33－1～33－2）が存在する。

プロセス管理テーブル31には、以下のような情報が置かれる。第1にそのプロセスを実行すべきプロセッサのプロセッサ番号を示すプロセッサ識別子321（321－1～321－4）である。実施例ではプロセス管理テーブル32－1のプロセッサ識別子は"1"となっており、このプロセス管理テーブルに対応するプロセス0（12－1）は、プロセッサ番号11－1に"1"を持つ、プロセッサ13－1で実行すべきことを示している。プロセスを実行するプロセッサが未定のときは、このプロセッサ識別子を0としておく。また、実行するプロセッサを特に決めず、どのプロセッサでもそのプロセスを実行できるようにしたい場合には、プロセッサ識別子を負の値にしておく。

第2に、プロセスの実行順を決定する優先度322（322－1～322－4）である。この優先度の数値が小さいほど、プロセスの優先度が高くなり、他のプロセスに優先して実行されなければならないことを示す。



第3は、実行待ち時間323（323－1～323－4）である。この実行待ち時間は、そのプロセスが実行待ち状態になっている時間を示す。プロセスが実行されるとこの実行待ち時間は0に戻される。

第4は、カウンタ324（324－1～324－4）である。このカウンタはプロセスの実行が妨絶された回数を数えるカウンタである。

第5は、プロセス固定フラグ325（325－1～325－4）である。ここには、プロセスをプロセッサに固定にするかどうかの情報が入る。固定としておくとそのプロセスは、プロセッサ識別子321に記されたプロセッサ以外では実行されない。

第6は、グループ識別子326（326－1～326－4）である。ここには、そのプロセスがどのグループに属しているかを示す情報が入る。グループの識別子とグループ管理テーブルは1対1に対応する。

グループ管理テーブルには、以下の情報が記されている。まず、第1にプロセッサ識別子である。これは、そのグループに属するプロセスを実行す



るプロセッサ番号が記されている。

第2は、実行待ち時間322（322－1～322－3）である。この実行待ち時間は、グループに属するプロセスのいずれもがプロセッサで実行されておらず、いずれかが実行待ち状態となっている時間である。グループのプロセスが1つでも実行されるとこの実行待ち時間は0に戻される。

第3は、カウンタである。このカウンタはグループに属するプロセスの実行が妨絶された回数を数えている。グループ内のプロセスが実行された時には0にクリアされる。

第4は、グループ固定フラグであり、このフラグが立っているときには、そのグループを実行するプロセッサは固定となる。

第3図は、オペレーティングシステムのプログラム31の一部であるプロセススケジューラ311のアルゴリズムを示している。

まず、実行待ち状態となっているプロセスの中から、優先順位に従い、次に実行するプロセスを選択する（311－1）。選択したプロセスのプロセス管理テーブルより、グループ識別子325を読みだす。このグループ識別子よりグループ管理テーブルを検索する（311－2）。

このグループ管理テーブル中のプロセッサ識別子331を読みだし、この値により以下の処理を行なう（311－3）。

その値が正の値だったならば、スケジューラプログラム311を実行しているプロセッサは、自分のプロセッサ番号を11（11－1～11－3）から読みだし、その正のプロセッサ識別子331の値と比較する（311－5）。一致していないときには、次に優先順位が高いプロセスを選択し（311－1）、一致しているときには選択したプロセスを自分のプロセッサに割り当てる（311－6）。

グループ管理テーブルのプロセッサ識別子が0だったときには、そのプロセッサ識別子の値を自分のプロセッサ番号（11）の値に変更して（311－4）、選択したプロセスを自分のプロセッサに割り当てる（311－6）。

プロセッサ識別子が負の値だったときには、他

続行に選択したプロセスを自分のプロセッサに割り当てる（311－6）。

第4図は、オペレーティングシステムのプログラム31の一部であるプロセススケジューラ311のアルゴリズムの別の実施例を示している。

まず、実行待ち状態となっているプロセスの中から、優先順位に従い、次に実行するプロセスを選択する（311－1）。選択したプロセスのプロセス管理テーブル32より、プロセッサ識別子321を読みだし、自分のプロセッサ番号11と比較する（311－7）。その値が等しかったときには、選択したプロセスを自プロセッサに割り当てる（311－6）。

等しくないときには、プロセス固定フラグ325が立っているかどうか判定する（311－8）。立っていた場合には、次に優先順位が高いプロセスを選択し直す（311－1）。

立っていないときには、プロセス管理テーブル32の実行待ち時間323がしきい値を超えていないか調べる（311－9）。



超えている場合には、プロセス管理テーブル32のプロセッサ識別子321を自プロセッサ番号に設定して（311－14）、選択したプロセスを自分に割り付ける（311－6）。

超えていない場合には、プロセス管理テーブル32中のグループ識別子326よりグループ管理テーブルを検索し、その中のプロセッサ識別子が自分のプロセッサ番号と等しいか調べる（311－10）。等しい場合には、やはり、311－14を実行する。

等しくない場合には、グループ管理テーブル33中のグループ固定フラグ334を調べる（311－11）。フラグが立っているときには、次に優先度の高いプロセスを調べる（311－1）。

フラグが立っていないときには、グループ管理テーブル33の実行待ち時間332がしきい値を超えていないか調べる。超えていない場合には、次に優先度の高いプロセスを調べる（311－1）。

超えていた場合には、グループ管理テーブル33のプロセッサ識別子331とプロセッサ管理テーブル32のプロセッサ識別子321を自分のプロセッサ



番号11に設定して（311－13，311－14）、選択したプロセスを実行する（311－6）。

第5図は、オペレーティングシステムのプログラム31の一部であるプロセススケジューラ311のアルゴリズムの別の実施例を示している。

まず、実行待ち状態となっているプロセスの中から、優先順位に従い、次に実行するプロセスを選択する（311－1）。選択したプロセスのプロセス管理テーブル32より、プロセッサ識別子321を読みだし、自分のプロセッサ番号11と比較する（311－7）。その値が等しかったときには、選択したプロセスを自プロセッサに割り当てる（311－6）。

等しくないときには、プロセス固定フラグ325が立っているかどうか判定する（311－8）。立っていた場合には、プロセス管理テーブル32のカウンタ324とグループ管理テーブル33のカウンタ333をインクリメントして（311－17，311－18）、次に優先順位が高いプロセスを選択し直す（311－1）。

プロセス固定フラグが、立っていないときには、



プロセス管理テーブル32のカウンタ325がしきい値を超えていないか調べる（311－15）。

超えている場合には、プロセス管理テーブル32のプロセッサ識別子321を自プロセッサ番号に設定して（311－14）、選択したプロセスを自分に割り付ける（311－6）。

超えていない場合には、プロセス管理テーブル32中のグループ識別子326よりグループ管理テーブルを検索し、その中のプロセッサ識別子が自分のプロセッサ番号と等しいか調べる（311－10）。等しい場合には、やはり、311－14を実行する。

等しくない場合には、グループ管理テーブル33中のグループ固定フラグ334を調べる（311－11）。フラグが立っているときには、311－17を実行する。

フラグが立っていないときには、グループ管理テーブル33のカウンタ333がしきい値を超えていないか調べる。超えていない場合には、311－17を実行する。

超えていた場合には、グループ管理テーブル33

のプロセッサ識別子331とプロセッサ管理テーブル32のプロセッサ識別子321を自分のプロセッサ番号11に設定して（311－13，311－14）、選択したプロセスを実行する（311－6）。

第6図は、第3図、第4図、第5図の311－6の選択したプロセスを自プロセッサに割り当てる方法を詳細に記したものである。

まず、今まで実行していたプロセスのレジスタを退避する（312－1）。これは、後にそのプロセスに実行の順番が再び廻ってきたときに、中断したところから再開できるようにするためである。

次に選択したプロセスのプロセス管理テーブル32の実行待ち時間323を0にする（312－2）。

同様に選択したプロセスが属しているグループのグループ管理テーブル33の実行待ち時間332、プロセス管理テーブル32のカウンタ324、グループ管理テーブルのカウンタ333も0にする（それぞれ、312－3，312－4，312－5）。

プロセスは、それぞれ独立なユーザ空間を持っているため、ユーザの空間を選択したプロセスのものに切り替える（312－6）。

最後に、プロセスを前回中断したときに退避しておいたレジスタの値を回復する（312－7）。退避するレジスタにはプログラムカウンタも含まれるため、前回中断した場所から処理が再開される。

第7図の313は、第1図14－1のタイマーがプロセッサに送る割り込みの処理ルーチンを示している。タイマー割り込みは10ms間隔で起こる。

タイマー割り込みが入るとプロセッサは、まず、プロセス管理テーブル32を全部調べていき、実行待ち状態となっているプロセスについては、その実行待ち時間323をインクリメントする（313－1）。

次にグループ管理テーブル33を全部調べていき、実行中のプロセスを含まず、実行待ちのプロセスを含むグループの待ち時間333をインクリメントする（313－2）。

第2図の本発明のシステム構成図の別の実施例を第8図に示す。第8図において、13－1～13－3はプロセッサ、11－1～11－3はプロセッサ番号、12－1～12－6はプロセス、14－2はタイマ





ー、4は接続バス、5は共有メモリ、31はＯＳプログラム、311はスケジューラ、32－1～32－4はプロセス管理テーブル、321－1～321－4はプロセッサ識別子、325－1～325－4はプロセス固定フラグ、326－1～326－4はグループ識別子、327－1～327－4はＣＰＵ時間、328－1～328－4，329－1～329－4，330－1～330－4は優先度、33－1～33－2はグループ管理テーブル、331－1～331－2はプロセッサ識別子、334－1～334－4はグループ固定フラグである。

プロセッサ13－1～13－3は、接続バス4により相互に、ならびに共有メモリ5に接続されている。各プロセッサはプロセッサ番号11－1～11－3をそれぞれ持っている。本実施例では、プロセッサ13－1～13－3はそれぞれプロセッサ番号"1"，"2"，"3"を有している。

プロセッサ13－1～13－3のうち、一つのプロセッサにはタイマー14－2があり、一定の時間おきにタイマー割り込みがそのプロセッサに入る。実施例では10msおきと1sおきの2種類の割り込みが入る。

各プロセスはプロセッサ上で実行される。共有メモリ5には、オペレーティングシステムのプログラム31、このプログラム31の一部であるプロセススケジューラ314、このオペレーティングシステムが管理するテーブルの一部であるプロセス管理テーブル32（32－1～32－3）、グループ管理テーブル33（33－1～33－2）が存在する。

プロセス管理テーブル32には、以下のような情報が書かれる。第1にそのプロセスを実行すべきプロセッサのプロセッサ番号を示すプロセッサ識別子21（321－1～321－4）である。実施例ではプロセス管理テーブル32－1のプロセッサ識別子は"1"となっており、このプロセス管理テーブルに対応するプロセス0（12－1）は、プロセッサ番号11－1に"1"を持つ、プロセッサ13－1で実行すべきことを示している。プロセスを実行するプロセッサが決定のときは、このプロセッサ識別子を0としておく。また、実行するプロセッサを明に決めず、どのプロセッサでもそのプロセスを

実行できるようにしたい場合には、プロセッサ識別子を負の値にしておく。

第2は、プロ1 ；固定フラグ325（325－1～325－4）である。ここには、プロセスをプロセッサに固定にするかどうかの情報が入る。固定としておくとそのプロセスは、プロセッサ識別子321に記される。

第3は、グループ識別子326（326－1～326－4）である。ここには、そのプロセスがどのグループに属しているかを示す情報が入る。グループの識別子とグループ管理テーブルは1対1に対応する。

第4は、ＣＰＵ時間である。これは、プロセスが使用したＣＰＵ時間であり、優先度の計算に使用する。ただし、このＣＰＵ時間は1秒毎に1／4にしているので正確なＣＰＵ時間ではない。

第5に、プロセスの実行順を決定する優先度328～330である。優先度が各プロセスに複数あるのは、プロセッサ毎に優先度を持っているためである。この優先度の数値が小さいほど、プロセスの優先度は高くなり、他の優先度の数値が大きいプロセスに優先して実行されなければならないことを示す。

グループ管理テーブルには、プロセッサ識別子331（331－1～331－2）がある。これに、そのグループに属するプロセスを実行するプロセッサ番号が記されている。

第9図は、第8図におけるスケジューラプログラム314のアルゴリズムを示している。スケジューラプログラムを実行しているプロセッサは、プロセス管理テーブルの優先度のうち自分に対応するものを順番に調べて行く。そして、優先度が最大（優先度の数字は最小）のプロセスを選択する（314－1）。ただし、このときプロセス管理テーブル32のプロセス固定フラグ325が立っており、プロセス識別子321が自分のプロセッサ番号でないプロセスは除く。

次に選択したプロセスを自分のプロセッサに割り当てる（314－2）。

第10図の316はタイマー14－2が発生させる10





msec毎の割り込みの処理ルーチンである。ここでは、各プロセッサが実行中のプロセスのプロセス管理テーブル32のＣＰＵ時間327をインクリメントしている。

第11図の316はタイマー14－2が発生させる1secごとの割り込みの処理ルーチンである。ここでは、各プロセス管理テーブル32の優先度328～330を再計算している。316－1で各プロセスについて処理を行なっており、316－3で各プロセッサについて処理を行なっている。316－2では、ＣＰＵ時間327が大きくなり過ぎないように1秒毎に1／4にしている。

316－4～316－7で優先度を計算している。優先度は、基本的にはそのプロセスのＣＰＵ時間に等しいが、プロセス管理テーブル32のプロセッサ識別子321に記されていないプロセッサの優先度には、20を加える。また、グループ管理テーブル33のプロセッサ識別子に書かれていないプロセッサの優先度にはやはり20を加える。

これにより、プロセス管理テーブルやグループ管理テーブルのプロセッサ識別子に書かれているプロセッサに対してはそのプロセスの優先度が高くなり、実行されやすくなる。

第12図は、シングルプロセッサ用に開発したプログラムを従来のスケジューリング方式のマルチプロセッサで実行した場合を示している。第12図において、12－1～12－2はプロセス、41－1～41－2は割り込み禁止処理、42－1～42－2は資源使用処理、43－1～43－2は割り込み許可処理である。

シングルプロセッサは、一瞬一瞬を見ると実行しているプロセスは一つである。よって、シングルプロセッサで実行することを前提としたプログラムは、共有資源を使用する場合にも、単に割り込みを禁止するだけで使用しているものが多い。これは、資源をロックして使用するより処理が高速なためである。シングルプロセッサでは割り込みさえ禁止しておけば、瞬間的には実行されているプロセスは一つなので資源を同時に複数のプロセスが使用するようなことはない。

しかし、マルチプロセッサでは第12図に示すように、41－1，41－2で割り込みを禁止しても、42－1と42－2のように同時に複数のプロセスが資源を使用することがある。

第13図は、本発明のスケジューリング方法を用いているマルチプロセッサでシングルプロセッサ用のプログラムを実行した場合を示す。同じ資源を使用するプロセス12－1と12－2を同一グループにして、プロセッサ0で実行するようにすれば、42－1と42－2の資源の使用が必ず直列に実行され、資源使用が競合することがない。

尚、近年、個々にプログラムとデータを持つ従来のプロセスに代わって、いくつか個にプログラムやデータを共有するスレッド（thread）やライト・ウエイト・プロセス(light weight process)が注目されている。

プログラムやデータを共有するスレッドでは共有データ量が多いため、グループ化したときのキャッシュ性能の向上が一段と大きくなる。

またプログラム，データを共有するスレッドをグループ化して、１つのプロセッサで実行すると、プログラム，データの切り替えがいらないことが多くなり、ＯＳの処理時間を短くすることができる。

従って、本発明は、かかるスレッドやライト・ウエイト・プロセスに対して適用しても効果大である。

【発明の効果】

本発明のプロセススケジューリング方法は、複数のプロセッサからなるマルチプロセッサシステムにおいて，複数のプロセッサのいずれでも実行可能なプロセスについて、あえてそのプロセスをグループ化してそのグループを実行させるプロセッサを特定のプロセッサに限定しようとするものである。

共有データを多く持つプロセスが異なるプロセッサで実行されると、キャッシュ一致化のための通信が頻繁に発生しキャッシュ性能を低下させる。本発明では、そのようなプロセスをグループ化し、同一のプロセッサで実行できるので、キャッシュ





一致化のための通信が不要となり、性能を向上させることができる。さらに、グループを特定のプロセッサに実行させることによりプロセッサ間の負荷が片寄ることがないように、一定時間以上実行されずに放置されているプロセスについては、いずれのプロセッサでも実行できるようにしており、負荷分散が可能となっている。ただし、この機能はプロセス固定フラグを立てることにより禁止することもできる。

また、本発明によって、排他制御を必要とするプロセス群を同一のプロセッサで動作させることが可能となり、共有メモリに対する排他制御を考慮していない従来の単一プロセッサ用の既存のプロセスを修正することなく使用しても正常な動作を保証することができる。この事は、他のプロセッサの不使用を意味するものではなく、前記プロセス群以外のプロセスについては、このような制約を設けずに、任意のプロセッサで実行可能としてマルチプロセッサの機能を有効に利用することができる。

## 4．図面の簡単な説明

第１図は本発明の一実施例のプロセススケジューリングの概要図、第２図はシステム構成図、第３図はスケジューラのアルゴリズム図、第４図はスケジューラアルゴリズムの別の実施例図、第５図もスケジューラのアルゴリズムの別の実施例図、第６図は選択したプロセスを自分のプロセッサに割り当てるアルゴリズム図、第７図はタイマー割り込みの処理ルーチン図、第８図はシステム構成の別の実施例図、第９図はスケジューラのアルゴリズム図、第10図は10ms毎のタイマー割り込みの処理ルーチン図、第11図は１秒毎のタイマー割り込みの処理ルーチン図、第12図は従来のスケジューリング方式でシングルプロセッサ用のプログラムを実行した場合の説明図、第13図は本発明のスケジューリング方法でシングルプロセッサ用のプログラムを実行した場合の説明図である。

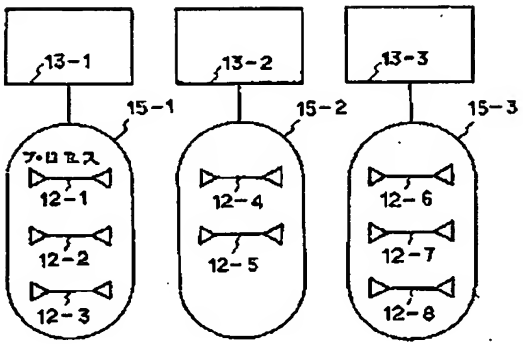
4…接続パス、5…共有メモリ、11…プロセッサ番号、12…プロセス、13…プロセッサ、14…タイマー、15…グループ、31…ＯＳプログラム、32

…プロセス管理テーブル、33…グループ管理テーブル、311…スケジューラ、321…プロセッサ識別子、322…優先度、323…実行待ち時間、324…カウンタ、325…プロセッサ固定フラグ、326…グループ識別子、327…ＣＰＵ時間、328～330…優先度、331…プロセッサ識別子、332…実行待ち時間、333…カウンタ、334…グループ固定フラグ、41…割り込み禁止処理、42…資源使用処理、43…割り込み許可処理。

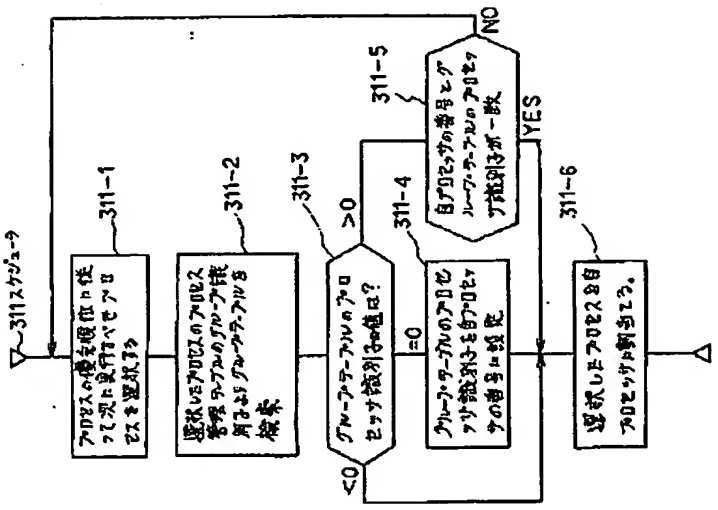
代理人　弁理士　　秋　本　正　実



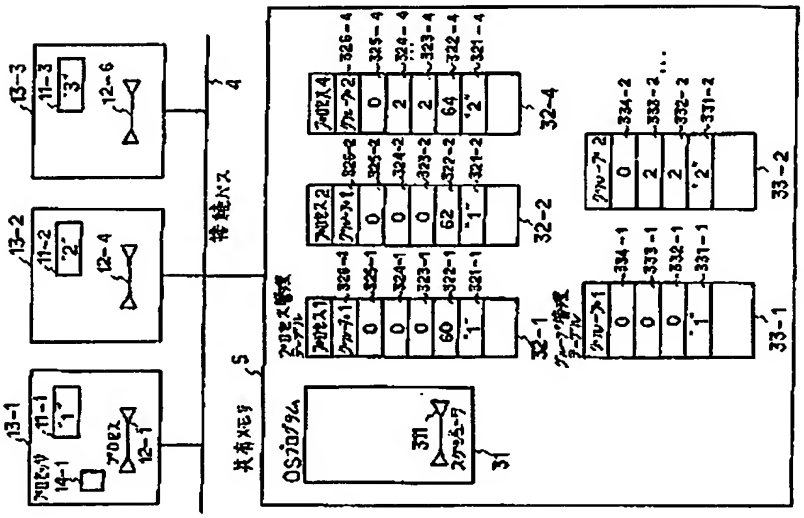
第　1　図



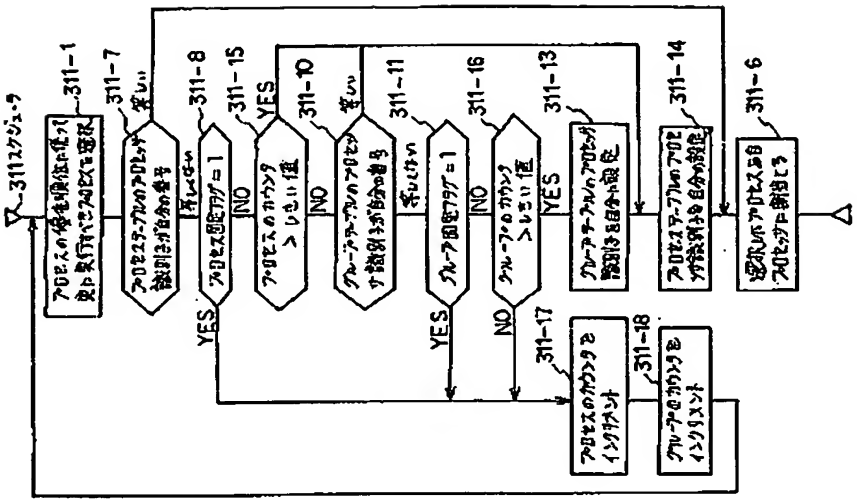
第 3 図



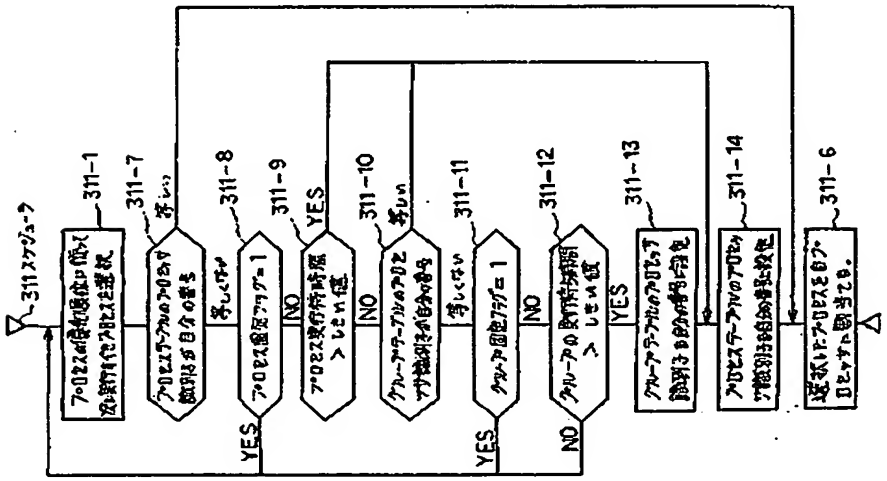

第 2 図



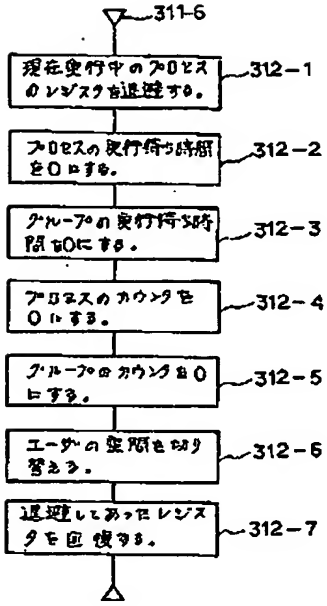
第 5 図



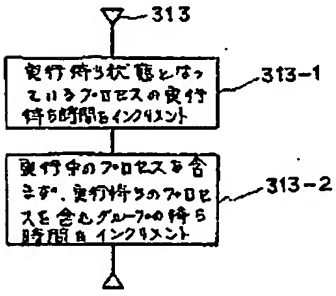
第 4 図



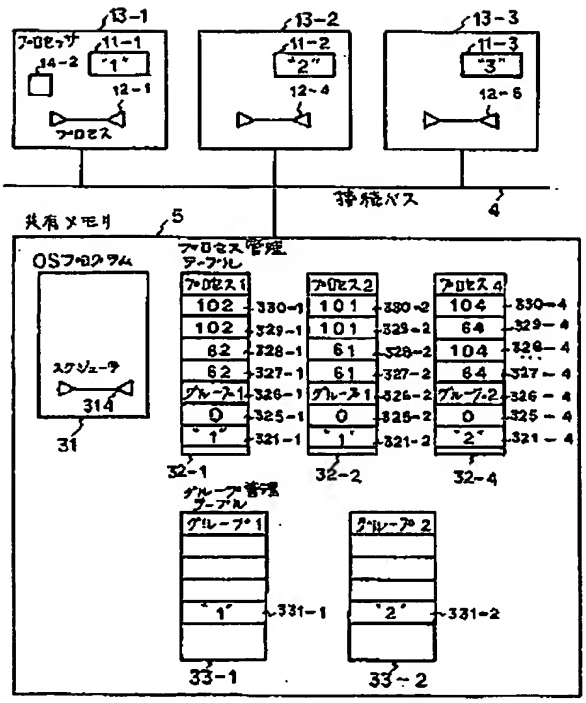
第 6 図



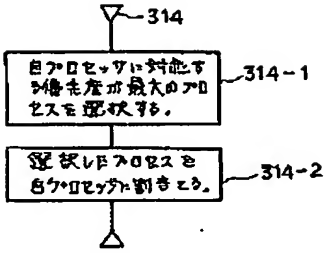

第 7 図



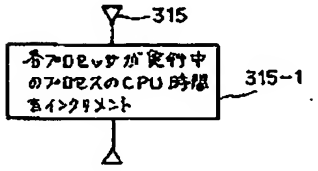

第 8 図



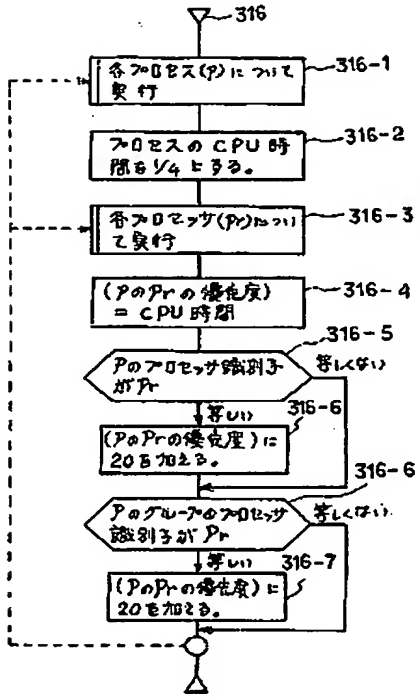
第 9 図



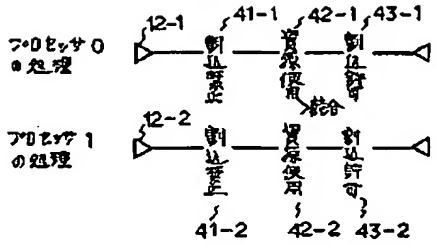

第 10 図



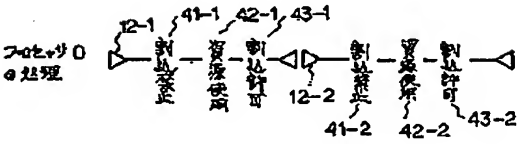

第 11 図



第 12 図




第 13 図




第1頁の続き
⑫発 明 者 中 村 智 明 茨城県日立市大みか町5丁目2番1号 株式会社日立製作所大みか工場内